



22883

PATENT TRADEMARK OFFICE

103.1043.01

1 This application is submitted in the name of the following inventors:

2	<u>Inventor</u>	<u>Citizenship</u>	<u>Residence City and State</u>
3	Gaurav Banga	India	Sunnyvale, CA
4	Henk J. Bots	United States	Hollister, CA
5	Mark Smith	United States	Cupertino, CA

6

7 The assignee is Network Appliance, Inc., a corporation having an office at 495  
East Java Dr., Sunnyvale, CA 94089.

8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
Title of the Invention

Prevention and Detection of IP Identification Wraparound Errors

Background of the Invention

15

16 *1. Field of the Invention*

17

18 This invention relates to reassembly of data fragments of fragmented datagrams in  
19 a communication system. In particular, the invention relates to reducing and/or detecting a  
20 likelihood of misassembly of data fragments in a communication system utilizing the Internet  
21 Protocol (IP) caused by IP identification wraparound.

## 2. *Description of the Related Art*

The Internet Protocol (IP) has become one of the most widely used communication protocols in the world. IP is part of a layered protocol, which means that another higher level protocol typically uses IP for data communication. Examples of such higher level protocols are the Transfer Control Protocol (TCP) and the User Datagram Protocol (UDP). In addition, even higher level protocols are sometimes utilized, such as the Network File System (NFS). These protocols are well known to those skilled in the art. The protocols are used to send data from a sending station (e.g., a client or a server on a sending end of a communication) to a receiving station (e.g., a client or a server on a receiving end of a communication), possibly through one or more routing devices that form an IP path.

In order to send a TCP, UDP or other protocol datagram across an IP connection, the datagram is encapsulated in an IP datagram. Often, the IP datagram must be fragmented into plural IP data fragments in order to be sent using the physical network. For example, if a size of the datagram exceeds the physical link's maximum transfer unit (MTU), that datagram must be fragmented into plural IP data fragments with sizes that do not exceed the MTU. Then, a receiving station reassembles the data fragments into the datagram.

A receiving station determines that data fragments belong to a single IP datagram by looking at an IP identification number in a header of each data fragment. All data fragments

1 from the same IP datagram share the same IP identification number. In addition, the header of  
2 each data fragment includes an offset from the start of the datagram, a length of the data  
3 fragment, and a flag that indicates whether or not the datagram includes more data fragments.  
4 This information is sufficient for reassembly of the IP datagram, which includes the original  
5 TCP, UDP or other protocol datagram.

6  
7 According to IP, the IP identification number is 16 bits long with a range of 0 to  
8 65535. A sending station conventionally uses a simple counter to determine the IP identification  
9 number for each IP datagram. In the early days of IP communications, a receiving station most  
10 likely would receive all data fragments of a datagram with a particular IP identification number  
11 and reassemble the datagram well before this counter could wrap around. If a data fragment was  
12 lost, thereby making reassembly of a datagram impossible, all received data fragments of that  
13 datagram would be discarded after a timeout of 64 seconds. With the slower communications  
14 times that existed in the early days of the IP communications, this timeout was usually sufficient  
15 to ensure data fragments would be discarded before the counter at the sending station could wrap  
16 around.

17  
18 However, today's Internet communications are much faster. Gigabit and 100Mb  
19 Ethernet implementations are commonplace, and faster implementations are constantly being  
20 developed. As the communications speed increases, the number of IP datagrams sent by a  
21 sending station per unit of time also increases. Thus, the simple 16-bit counter conventionally

1 used to generate IP identification numbers wraps around much more quickly. In fact, in a high  
2 speed setting, the counter can almost be guaranteed to wrap around within 64 seconds. Thus, a  
3 receiving station can receive data fragments from two different IP datagrams that share a  
4 common IP identification number before a first one of those datagrams is reassembled.

5  
6 Because of the nature of IP communications, it is possible for a data fragment  
7 from a second one of two datagrams to arrive at the receiving station before a corresponding data  
8 fragment from a first one of the two datagrams. Then, if the two datagrams share a common IP  
9 identification number due to wraparound of the sending station's IP identification number  
10 counter, the receiving station can misassemble the data fragments. This misassembly can result  
11 in corruption of the datagram.

12  
13 For example, if first datagram A is fragmented into data fragments A1, A2, A3,  
14 A4 and A5, and second datagram B is fragmented into data fragments B1, B2, B3 and B4, it is  
15 possible for a receiving station to receive the data fragment B2 before data fragment A2. Then, if  
16 datagram A and datagram B share a common IP identification number due to wraparound of the  
17 sending station's IP identification number counter, the receiving station can misassemble data  
18 fragments A1, B2, A3, A4 and A5 into a datagram, which of course would not contain the proper  
19 data.

1 Higher level protocols such as TCP and UDP utilize checksums and length checks  
2 in an attempt to catch such data corruption. However, the UDP checksum is only 16 bits long. It  
3 has been found that in a high speed environment, IP misassembly errors might occur with  
4 sufficient frequency that eventually a "false positive" checksum can result. In this case, the  
5 checksum can indicate that the UDP datagram has been properly reassembled, while in fact the  
6 datagram has been corrupted. Other properties of conventional IP exacerbate this situation, such  
7 as IP's acceptance of overlapping data fragments during datagram reassembly. In a UDP  
8 communication setting, these types of errors can lead to undetected data corruption. This data  
9 corruption might only come to light when the data is actually utilized, a situation that preferably  
10 should be avoided.

#### Summary of the Invention

11 The invention addresses the foregoing concerns by implementing measures  
12 designed to reduce a likelihood of misassembly of received data fragments from fragmented IP  
13 datagrams. In addition, the invention implements measures designed to detect when a likelihood  
14 of such misassembly is high so that appropriate corrective policies can be implemented.

15 One embodiment of an aspect of the invention is a method of generating IP  
16 identification numbers for IP datagrams. In this embodiment, a plurality of IP identification  
17 number generators is maintained. A plurality of receiving stations are associated with the

1 plurality of IP identification number generators such that each receiving station has an IP  
2 identification number generator associated therewith. An IP identification number for a  
3 datagram sent to one of the receiving stations is generated based on an output of the associated IP  
4 identification number generator. This method preferably is performed by an IP layer of a sending  
5 station's communication system.

6  
7 By using plural number generators, this aspect of the invention slows down  
8 wraparound of IP identification numbers used for communication with any given receiving  
9 station.

10  
11 Preferably, each of the IP identification number generators has at least one  
12 receiving station associated therewith. At least one of the IP identification number generators  
13 preferably has plural receiving stations associated therewith. In one embodiment, the plurality of  
14 IP identification number generators forms an array of number generators such as 16-bit counters.  
15 Preferably, the plurality of IP identification number generators is associated with the plurality of  
16 receiving stations by hashing destination addresses for the receiving stations and, in one  
17 embodiment, protocols for transmitting to those receiving stations so as to form an index to the  
18 array. If the hashing includes protocol information, the hashing preferably is performed such that  
19 at least half of the number generators in the array are associated with UDP protocol  
20 communications.

1           An embodiment of another aspect of the invention is a method of reducing a  
2       likelihood of misassembly of data fragments from fragmented IP datagrams. In this method, data  
3       fragments of a datagram having an IP identification number are received. All received data  
4       fragments of the datagram are discarded upon detection of receipt of an overlapping data  
5       fragment having the IP identification number, wherein the overlapping data fragment overlaps  
6       data in an already-received data fragment. The overlapping data fragment can overlap all or less  
7       than all of the already-received data fragment(s). This method preferably is performed by an IP  
8       layer of a receiving station's communication system.

9           An embodiment of another aspect of the invention also is a method of reducing a  
10      likelihood of misassembly of data fragments from fragmented IP datagrams. According to this  
11      method, a timeout for reassembling the datagrams is reduced to less than a standard timeout.  
12      Preferably, the datagram reassembly timeout is reduced to 45 seconds from the standard timeout  
13      of 64 seconds. Alternatively, the datagram reassembly timeout is dynamically reduced based on  
14      NFS data for round-trip times between a sending station and a receiving station. This method  
15      preferably is performed by an IP layer of a receiving station's communication system.

16  
17  
18           Yet another aspect of the invention is embodied in a method of reducing a  
19      likelihood of misassembly of data fragments from fragmented IP datagrams. This method  
20      includes the steps of receiving data fragments of a datagram having an IP identification number,  
21      and reducing a remaining time for reassembling the datagram upon detection of a gap in the

1 received data fragments. Preferably, the remaining time for reassembling the datagram is  
2 reduced to eight seconds. This method also preferably is performed by an IP layer of a receiving  
3 station's communication system.

4  
5 An additional aspect of the invention is embodied in another method of reducing a  
6 likelihood of misassembly of data fragments from fragmented IP datagrams. According to this  
7 method, data fragments of a first datagram are received, with the data fragments each having a  
8 protocol identification number, a source address, and a first IP identification number. A  
9 remaining time for reassembling the datagram is reduced upon detection, before receipt of a last  
10 data fragment of the first datagram, of a data fragment of a second datagram having the protocol  
11 identification number and the source address but having a second IP identification number.  
12 Preferably, the remaining time for reassembling the datagram is reduced to eight seconds. This  
13 method also preferably is performed by an IP layer of a receiving station's communication  
14 system.

15  
16 A further aspect of the invention is embodied in a method of detecting a  
17 likelihood of misassembly of data fragments from fragmented IP datagrams. In this embodiment,  
18 communication errors between a sending station and a receiving station are detected. The  
19 likelihood of misassembly is determined to be high upon detection that the communication errors  
20 occur at a high rate for a predefined period of time. The communication errors that are detected  
21 can include communication errors detected by an IP layer of the receiving station's



1 communication system. Such IP communication errors include, but are not limited to, receipt of  
2 overlapping data fragments and IP datagram reassembly timeout errors. The communication  
3 errors that are detected also can include communication errors detected by a UDP layer of the  
4 receiving station's communication system. Such UDP communication errors include, but are not  
5 limited to, UDP length errors and UDP checksum errors. The communication errors that are  
6 detected also can include communication errors detected by an NFS layer of the sending station's  
7 communication system.

8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1940  
1941  
1942  
1943  
1944  
1945  
1946  
1947  
1948  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979  
1980  
1981  
1982  
1983  
1984  
1985  
1986  
1987  
1988  
1989  
1990  
1991  
1992  
1993  
1994  
1995  
1996  
1997  
1998  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066  
2067  
2068  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2080  
2081  
2082  
2083  
2084  
2085  
2086  
2087  
2088  
2089  
2090  
2091  
2092  
2093  
2094  
2095  
2096  
2097  
2098  
2099  
2100  
2101  
2102  
2103  
2104  
2105  
2106  
2107  
2108  
2109  
2110  
2111  
2112  
2113  
2114  
2115  
2116  
2117  
2118  
2119  
2120  
2121  
2122  
2123  
2124  
2125  
2126  
2127  
2128  
2129  
2130  
2131  
2132  
2133  
2134  
2135  
2136  
2137  
2138  
2139  
2140  
2141  
2142  
2143  
2144  
2145  
2146  
2147  
2148  
2149  
2150  
2151  
2152  
2153  
2154  
2155  
2156  
2157  
2158  
2159  
2160  
2161  
2162  
2163  
2164  
2165  
2166  
2167  
2168  
2169  
2170  
2171  
2172  
2173  
2174  
2175  
2176  
2177  
2178  
2179  
2180  
2181  
2182  
2183  
2184  
2185  
2186  
2187  
2188  
2189  
2190  
2191  
2192  
2193  
2194  
2195  
2196  
2197  
2198  
2199  
2200  
2201  
2202  
2203  
2204  
2205  
2206  
2207  
2208  
2209  
2210  
2211  
2212  
2213  
2214  
2215  
2216

1 predetermined rate is once every ninety seconds. Alternatively, NFS re-transmissions also are  
2 considered when determining if a likelihood of datagram misassembly is high.

3  
4 Policies are preferably implemented to reduce the likelihood of misassembly of  
5 data fragments upon determining that the likelihood of misassembly is high. Examples of such  
6 policies include, but are not limited to, preferentially using TCP instead of UDP, using additional  
7 checksums, and presenting a warning message to a system administrator. When the sending  
8 station maintains plural IP identification number generators, such policies also can include  
9 reducing a number of receiving stations associated with the IP identification number generator  
10 that is wrapping around at faster than the predetermined rate.

11  
12 Each of the foregoing methods can be used in conjunction with the others in  
13 various combinations to reduce and/or to detect a likelihood of misassembly of IP datagrams.  
14 The invention also includes apparatuses such as sending and receiving stations configured to  
15 perform the foregoing methods, computer readable code by itself or embodied in a computer  
16 program product to cause a computer to perform the foregoing methods, and a memory storing  
17 information including instructions executable by a processor to perform the foregoing methods.

18  
19 This brief summary has been provided so that the nature of the invention may be  
20 understood quickly. A more complete understanding of the invention may be obtained by

reference to the following description of the preferred embodiments thereof in connection with the attached drawings.

#### Brief Description of the Drawings

Figure 1 is a representational view of communication between a sending station and a receiving station across a network such as the Internet.

Figure 2 is a representational view of a sending station using plural identification number generators to generate IP identification numbers.

Figure 3 is a representational view of a receiving station discarding a datagram upon detection of an overlapping data fragment.

Figure 4 is a representational view of a receiving station discarding a datagram upon detection of a partially overlapping data fragment.

Figure 5 is a representational view of a reduced timeout for reassembling datagrams at a receiving station.

1           Figure 6 is a representational view of a receiving station reducing a remaining  
2 time for reassembling a datagram upon detection of a gap in received data fragments of the  
3 datagram.

4  
5           Figure 7 is a representational view of a receiving station reducing a remaining  
6 time for reassembling a datagram upon detection of a data fragment from another datagram  
7 having that same source address and protocol as the datagram but a different IP identification  
8 number.

9  
10           Figure 8 is a flowchart for explaining determination that a likelihood of  
11 misassembly of datagrams is high upon detection of a high rate of communication errors for a  
12 period of time.

13  
14           Figure 9 is a flowchart for explaining determination that a likelihood of  
15 misassembly of datagrams is high upon determination that an IP identification number generator  
16 wraps around at faster than a predetermined rate.

#### 17 18           Description of the Preferred Embodiment

19  
20           In the following description, a preferred embodiment of the invention is described  
21 with regard to preferred process steps and data structures. However, those skilled in the art

1 would recognize, after perusal of this application, that embodiments of the invention may be  
2 implemented using one or more general purpose processors or special purpose processors  
3 adapted to particular process steps and data structures operating under program control, that such  
4 process steps and data structures can be embodied as information stored in or transmitted to and  
5 from memories (e.g., fixed memories such as DRAMs, SRAMs, hard disks, caches, etc., and  
6 removable memories such as floppy disks, CD-ROMs, data tapes, etc.), with the information  
7 including instructions executable by such processors (e.g., object code that is directly executable,  
8 source code that is executable after compilation, code that is executable through interpretation,  
9 etc.), and that implementation of the preferred process steps and data structures described herein  
10 using such equipment and structures would not require undue experimentation or further  
11 invention.

12  
13  
14 Fig. 1 is a representational view of communication between a sending station and  
15 a receiving station across a network such as the Internet. In Fig. 1, sending station 1 sends  
16 information across network 2 to receiving station 3.

17 Sending station 1 can be a client sending data to a server, a server sending data to  
18 a client, or any other device or entity sending data across network 2. Likewise, receiving station  
19 3 can be a server receiving data from a client, a client receiving data from a server, or any other  
20 device or entity receiving data across network 2.  
21

1           A single device, such as a client or a server, can be both a sending station and a  
2 receiving station, possibly simultaneously. For example, in typical two-way data  
3 communications between a client and a server, the client is a sending station for communications  
4 sent to the server and a receiving station for communications received from the server. Likewise,  
5 the server is a receiving station for communications received from the client and a sending  
6 station for communications sent to the client.

7  
8           Sending station 1 communicates through a layered communication protocol.  
9 Preferably, the layered communication protocol includes application layer 5, higher level layer 6  
10 such as a Network File System (NFS) layer, transport layer 7 such as a Transfer Control Protocol  
11 (TCP) layer, User Datagram Protocol (UDP) or other protocol layer, and Internet Protocol (IP)  
12 layer 8. Various other combinations of layers are possible. For example, some sending stations  
13 do not have higher level layer 6. Also, particular types of layers are designed to work with other  
14 types of layers. For example, NFS was originally designed to work with UDP, not TCP. Finally,  
15 some applications directly utilize the lower level UDP or IP layers, thereby bypassing much of  
16 the error checking (e.g., checksum computations) provided by the higher level and application  
17 layers. As data passes through each of the layers from an application program, each layer  
18 performs operations on the data such as encapsulation.

19  
20           Application layer 5 provides an interface for application programs to send data.  
21 Application layer 5 might compute and add a checksum to the data. Such a checksum is useful

1 for ensuring data integrity at a receiving station. However, the application layer does not have to  
2 use any such checksum.

3  
4 Higher level layer 6 such as an NFS layer typically keeps track of network data.  
5 This layer also can add a checksum, although such is not mandatory.

6  
7 Transport layer 7 packages data in datagrams. Each datagram typically includes a  
header and data. The data may be of various lengths. The header typically includes source and  
address information, datagram length, and a checksum. For example, UDP specifies that a UDP  
datagram has a header with a 16 bit source port number, a 16 bit destination port number, a 16 bit  
UDP length, and a 16 bit checksum. The checksum is for both the datagram's header and data,  
as well as for a pseudo-header that includes additional information (IP source address, IP  
destination address, protocol, and datagram length).

15 IP layer 8 encapsulates UDP, TCP or other protocol datagrams into IP datagrams  
16 in order to send those datagrams across network 2. Often, an IP datagram must be fragmented  
17 into plural IP data fragments in order to be sent across network 2. For example, if a size of the  
18 datagram exceeds a known maximum transfer unit (MTU) for network 2, that datagram must be  
19 fragmented into plural IP data fragments with sizes that do not exceed the MTU.

1 IP layer 8 generates an IP identification number for each IP datagram. All data  
2 fragments from the same IP datagram share the same IP identification number. In addition, the  
3 header of each data fragment includes an offset from the start of the datagram, a length of the  
4 data fragment, and a flag that indicates whether or not the datagram includes more data  
5 fragments. This information is sufficient for a receiving station to reassemble the IP datagram,  
6 which includes the original TCP, UDP or other protocol datagram. IP datagrams also include a  
7 checksum, but only for the header information.

8  
9  
10  
11  
12  
13  
14  
15 According to IP, the IP identification number is 16 bits long with a range of 0 to  
16 65535. In a high-speed communications setting, a conventional sending station might send many  
17 more than 65535 datagrams in a short period of time, causing this IP identification number to  
18 wrap around quickly. Thus, a sending station might send data fragments from two different  
19 datagrams with the same IP identification number to the same receiving station. This duplicate  
20 IP identification number can cause the receiving station to try to misassemble some of these data  
21 fragments into a single datagram.

17 In a setting where only a UDP data checksum is used to verify data integrity (e.g.,  
18 an application checksum is not used or is bypassed and TCP is not used), some of these  
19 misassembled datagrams can slip through the weak 16 bit UDP checksum. This problem is  
20 exacerbated by the fact that the UDP checksum's strength is data-type dependent, resulting in  
21 similar checksums for similar types of data. For example, a corrupt datagram resulting from



1 misassembly of mismatched text data fragments has better than a 1:65535 chance of resulting in  
2 a checksum that matches the checksum for the original datagram.

3  
4 In order to reduce a likelihood of wraparound of IP identification numbers for  
5 datagrams sent to a particular receiving station, an IP layer according to the invention utilizes  
6 plural IP identification number generators, as discussed below with respect to Fig. 2.

7  
8 In order to help determine when a likelihood of misassembly of datagrams is high,  
9 a sending station according to the invention can monitor for a high rate of communication errors  
10 that might be the result of datagram misassembly, as discussed below with reference to Fig. 8.  
11 The sending station also can monitors the IP identification number generator(s) for rapid  
12 wraparound, as discussed below with reference to Fig. 9.

13  
14 Returning to Fig. 1, network 2 preferably includes a plurality of routers 10.  
15 Examples of network 2 include the Internet, an intranet, an Ethernet network, and any other  
16 network or virtual network that utilizes IP communications. The particular configuration of  
17 network 2 is representational only of the inclusion of many routers and many possible  
18 communication paths through network 2. This configuration has no other significance, and any  
19 other configuration that allows communications through network 2 can be utilized with the  
20 invention. For example, network 2 could be replaced with a single router 10 between sending  
21 station 1 and receiving station 3.

Each of routers 10 can have an MTU smaller than the sizes of data fragments sent to that router. If a router receives a data fragment larger than the router's MTU, the router can further fragment the data fragment. Each data fragment of a datagram can take a different path through network 2. The routers along these different paths can have different MTUs. Thus, data fragments of a single datagram received by a receiving station can have different sizes.

Receiving station 3 also communicates through a layered communication protocol. Preferably, the layered communication protocol includes layers corresponding to layers in sending stations that might send data to the receiving station. Thus, in Fig. 1, the layered communication protocol of receiving station 3 includes IP layer 12, transport layer 13 such as a TCP, UDP or other protocol layer, higher level layer 14 such as an NFS layer, and application layer 15. Various other combinations of layers are possible, and some applications directly utilize the lower level UDP or IP layers, thereby bypassing much of the error checking (e.g., checksum verifications) provided by the higher level and application layers. As data passes through each of the layers to an application program, each layer performs operations on the data such as decapsulation.

IP layer 12 reassembles data fragments into datagrams based on IP identification numbers, length data and flags in the headers of those data fragments. Reassembly time for a datagram is limited by a timeout. If datagram reassembly time from when a first data fragment of

1 a datagram is received exceeds the timeout, all data fragments associated with the datagram are  
2 discarded.

3  
4 IP layer 12 also verifies a header checksum for received unfragmented datagrams  
5 and data fragments, but this checksum only verifies the integrity of the associated IP headers.  
6 This checksum therefore does not generally help prevent or detect data fragment misassembly, at  
7 least because such misassembly can occur with completely self-consistent IP headers.

8  
9  
10 In order to reduce a likelihood of misassembly of data fragments from different  
11 datagrams that have the same IP identification number, an IP layer of a receiving station  
12 according to the invention can take several actions. The IP layer can discard all data fragments of  
13 a datagram if an overlapping data fragment is received, as discussed below with reference to  
14 Figs. 3 and 4. The IP layer also can reduce a timeout for datagram reassembly. The overall  
15 timeout can be reduced, as discussed below with reference to Fig. 5. In addition, the time for  
16 reassembly can be dynamically reduced if a gap is detected in received data fragments of a  
17 datagram, as discussed below with respect to Fig. 6, or if a data fragment from another datagram  
18 with a different IP identification number is received from the same source with the same  
19 protocol, as discussed below with respect to Fig. 7.

20 In order to help determine when a likelihood of misassembly of datagrams is high,  
21 a receiving station according to the invention can monitor for a high rate of communication

1 errors that might be the result of datagram misassembly, as discussed below with reference to  
2 Fig. 8.

3  
4 Transport layer 8 strips the TCP, UDP or other protocol header off of a datagram,  
5 as appropriate. Both TCP and UDP can verify a checksum for the resulting data. However, as  
6 noted above, the UDP checksum is relatively weak. It should be noted that the TCP checksum  
7 also is not perfect. Corrupt data sometimes passes the TCP checksum, albeit with significantly  
8 less frequency than with the UDP checksum.

9  
10 The length of the datagram preferably also is verified by the transport layer.  
11 However, many length errors are corrected in the IP layer's datagram reassembly. For example, a  
12 750 byte data fragment inserted into a space for a 500 byte data fragment during datagram  
13 reassembly typically will not result in a UDP or TCP length error because the IP layer truncates  
14 overlong data fragments. Thus, length error checking also may not help catch datagram  
15 misassembly.

16  
17 Higher level layer 14 preferably works in conjunction with higher level layer 6 in  
18 sending station 1 to keep track of and to manage network data. Higher level layer 14 also can  
19 provide data integrity verification through checksums, although use of such checksums is not  
20 mandatory.

1           Application layer 15 provides an interface for an application program to receive  
2 data. This layer also optionally can provide a checksum and error checking.

3  
4           As is evident from the discussion above, many layers of an layered protocol used  
5 for network communications can provide checksums and other error detection measures.  
6 However, one common method for network communications is to have application programs  
7 directly communicate using UDP and IP. The only data checksum in this configuration is the  
8 UDP checksum, which is weak enough that it might miss some misassembly of data fragments.  
9 The invention provides techniques for decreasing the likelihood of such misassembly, as well as  
10 for detecting when a likelihood of misassembly is high.  
11  
12  
13  
14

15           Fig. 2 is a representational view of a sending station using plural identification  
16 number generators to generate IP identification numbers.

17           Briefly, IP identification numbers for IP datagrams are generated. To generate  
18 these identification numbers, a plurality of IP identification number generators are maintained. A  
19 plurality of receiving stations are associated with the plurality of IP identification number  
20 generators such that each receiving station has an IP identification number generator associated  
therewith. An IP identification number is generated for a datagram sent to one of the receiving  
stations based on an output of the associated IP identification number generator. Preferably, the

1 IP identification numbers are generated in an IP layer of a sending station's communication  
2 system.

3  
4 In more detail, Fig. 2 shows sending station 1 with array 17 of N plural IP  
5 identification number generators 18. N preferably is a power of two to simplify indexing and  
6 hashing, which are discussed below. Examples of N are 16 and 256. Each of IP identification  
7 number generators 18 preferably is a 16-bit counter, corresponding to the 16 bits needed for an IP  
8 identification number.

9  
10 In order to associate a receiving station with an IP identification number  
11 generator, sending station 1 preferably uses the receiving station's address. Optionally, sending  
12 station 1 also uses the protocol for a particular datagram to be sent to that receiving station.  
13 Preferably, the datagram's transport protocol (i.e., TCP, UDP or other protocol) is used for this  
14 protocol. As shown in Fig. 2, receiving station address and protocol 20 for a datagram are  
15 hashed by hash 21 to form index 22 to array 17.

16  
17 In the preferred embodiment, there are more than N possible combinations of  
18 receiving station addresses and protocols. In fact, sending station 1 may send data to more than  
19 N separate receiving stations. Therefore, more than one receiving station can be associated with  
20 each of IP identification number generators 18.

1 Hash 21 preferably is designed so that IP identification number generators 18 are  
2 distributed fairly evenly among the receiving stations. Preferably, if there are more than N  
3 receiving stations, each of plural IP identification number generators 18 has at least one receiving  
4 station associated therewith.

5  
6 Furthermore, because UDP tends to be more susceptible to datagram misassembly  
7 than other transport protocols, hash 21 preferably is designed so that half of IP identification  
8 number generators 18 are associated with UDP. The other half of IP identification number  
9 generators 18 preferably are associated with all other protocols. Thus, each receiving station  
10 preferably will have an IP identification number generator associated therewith for UDP  
11 datagrams and an IP identification number generator associated therewith for all other protocol  
12 datagrams. This feature of hash 21 can be implemented by including a "UDP/non-UDP" bit in  
13 hash 21.

14  
15 Whenever sending station 1 needs to send an IP datagram to a receiving station,  
16 sending station 1 preferably sends receiving station address and protocol 20 for that datagram  
17 through hash 21 to form index 22. Index 22 is then used to index to one of the plural IP  
18 identification number generators 18, which provides the identification number and then  
19 increments (or vice versa).

By virtue of the foregoing arrangement, a single IP identification number generator is not shared among all receiving stations. Rather, each of plural IP identification number generators is shared only among the associated receiving station/protocol combinations. Wraparound of IP identification numbers for datagrams sent to a particular receiving station using a particular protocol thereby tends to be greatly slowed, reducing a likelihood that data fragments from two datagrams having the same IP identification number will be sent to the same receiving station before reassembly timeout.

Figs. 3 to 6 are representational views that illustrate various techniques by which a receiving station can further reduce a likelihood of misassembly of data fragments from two different datagrams. In each of these figures, a data fragment is represented by a small box. A letter in the box represents a datagram to which the data fragment belongs, and a number in the box represents the data fragment's position in the datagram. A small numeral to the upper right of each box indicates an order in which the data fragments have been received by the receiving station in each illustrated example. A data fragment that has not been received does not have such a numeral and is designated by a broken outline (see, e.g., data fragment A2 in Fig. 3). Finally, a numeral under each data fragment indicates a size of the data fragment in bytes.

It should be noted that the details shown in Figs. 3 to 6, such as specific orders, sizes and compositions of data fragments and datagrams, are provided solely to clarify aspects of the invention discussed with respect to each figure and are for illustrative purposes only. The



1 invention is in no way limited to those particular details, as will be apparent to one skilled in the  
2 art.

3  
4 Fig. 3 is a representational view of a receiving station discarding a datagram upon  
5 detection of an overlapping data fragment.

6  
7 Briefly, a likelihood of misassembly of data fragments from fragmented IP  
8 datagrams is reduced. Data fragments of a datagram having an IP identification number are  
9 received. All received data fragments of the datagram are discarded upon detection of receipt of  
10 an overlapping data fragment having the IP identification number, wherein the overlapping data  
11 fragment overlaps data in an already-received data fragment. Preferably, this technique is  
12 performed by an IP layer of a receiving station's communication system.

13  
14 In more detail, receiving station 3 in Fig. 3 has received data fragments A1, A3,  
15 A4 and A5. Data fragment A2 has not been received. Subsequently, data fragment B1 has been  
16 received. Datagrams A and B have identical IP identification numbers in Fig. 3, for example as a  
17 result of an IP identification number generator wrapping around in a sending station that sent  
18 datagrams A and B. Thus, data fragment B1 overlaps data fragment A1. In other words, if data  
19 fragment B1 was assembled into a datagram with data fragment A1, some data from one of the  
20 data fragments would overlap data from the other data fragment. This overlapping corresponds

1 to a situation where misassembly can occur, for example if a data fragment B2 was subsequently  
2 received. In Fig. 3, data fragment B1 overlaps all of data fragment A1.

3  
4 According to the invention, receiving station 3 determines that data fragment B1  
5 has overlapped data fragment A1. Upon determining that such overlapping has occurred, a  
6 receiving station according to the invention discards all received data fragments for the datagram  
7 with the overlapped data fragment. In Fig. 3, the datagram with overlapped data fragment A1 is  
8 datagram A, so data fragments A1, A3, A4 and A5 are discarded. The invention similarly would  
9 have discard all received data fragments for datagram A if another of its data fragments had been  
10 overlapped instead of data fragment A1, for example data fragment A3, A4 or A5.

11 By virtue of the foregoing operation, a sending station discards data fragments  
12 before misassembly can occur in some situations.  
13  
14

15 Fig. 4 is a representational view of a receiving station discarding a datagram upon  
16 detection of a partially overlapping data fragment. In Fig. 4, receiving station 3 has received data  
17 fragments A1, A3, A4 and A5. Data fragment A2 has not been received. Subsequently, data  
18 fragment B2 has been received. Data fragment B2 has a size of 750 bytes, versus the 500 byte  
19 size of data fragments A1, A3, A4 and A5. Such a difference in data fragment size can occur, for  
20 example, if data fragment B2 traveled across network 2 along a path that had an MTU of 750

1 bytes, while the rest of the data fragments traveled across network 2 along paths with MTUs of  
2 500 bytes.

3  
4 Datagrams A and B have identical IP identification numbers in Fig. 4. In this  
5 situation, the first 500 bytes of data fragment B2 do not overlap any received data fragments of  
6 datagram A. However, the last 250 bytes of data fragment B2 do overlap part of data fragment  
7 A3. A receiving station according to the invention preferably would detect this overlap and  
8 would therefore discard data fragments A1, A3, A4 and A5. Thus, the invention preferably  
9 discards data fragments of a datagram when any data in any of those data fragments is overlapped  
10 by any data in a subsequently received data fragment with the same IP identification number.  
11

12 By virtue of the foregoing operation, a receiving station discards data fragments  
13 before misassembly can occur in more situations than if only overlap of entire data fragments  
14 was acted upon.  
15

16 Fig. 5 is a representational view of a reduced timeout for reassembling datagrams  
17 at a receiving station.

18  
19 Briefly, a likelihood of misassembly of data fragments from fragmented IP  
20 datagrams is reduced by reducing a timeout for reassembling the datagrams to less than a

1 standard timeout. Preferably, this technique is performed by an IP layer of a receiving station's  
2 communication system.

3  
4 In more detail, receiving station 3 in Fig. 5 has received data fragments A1 and  
5 A3 of datagram A and data fragment B2 of datagram B. Data fragment A2 has not been  
6 received. Datagrams A and B have identical IP identification numbers. Accordingly, data  
7 fragments A1, B2 and A3 could be misassembled into a corrupted datagram as long as the data  
8 fragments were all received before a timeout for datagram reassembly using data fragments A1  
9 and A3.  
10  
11

12 In Fig. 5, time line 23 illustrates a standard IP datagram assembly timeout of 64  
13 seconds. Data fragment B2 is received within this time frame, so if this standard timeout was  
14 used, misassembly could occur. However, a receiving station according to the invention  
15 preferably uses a reduced timeout such as that illustrated by time line 24. This timeout ends  
16 before receipt of data fragment B2, preventing any chance of misassembly in the example shown  
17 in Fig. 5.

18 A timeout of 45 seconds has been found to produce good result in terms of  
19 allowing enough time for proper datagram reassembly while preventing some datagram  
20 misassembly. Alternatively, the timeout could be determined based on network data for expected  
21 communication (e.g., round-trip) times between a particular sending station and a particular

1 receiving station. Such network data preferably could be provided by an NFS layer of each  
2 station's communication system.

3  
4 Fig. 6 is a representational view of a receiving station reducing a remaining time  
5 for reassembling a datagram upon detection of a gap in received data fragments of the datagram.

6  
7 Briefly, a likelihood of misassembly of data fragments from fragmented IP  
8 datagrams is reduced. Data fragments of a datagram having an IP identification number are  
9 received. A remaining time for reassembling the datagram is reduced upon detection of a gap in  
10 the received data fragments. Preferably, this technique is performed by an IP layer of a receiving  
11 station's communication system.

12  
13 In more detail, receiving station 3 in Fig. 6 has received data fragments A1 and  
14 A3 of datagram A. Data fragment A2 has not been received, creating a gap in the received data  
15 fragments. The gap can be detected by examining the lengths and offsets included in the headers  
16 of the received data fragments. This gap indicates that data fragment A2 might have been lost in  
17 transit, opening up the opportunity for a data fragment from another datagram to be improperly  
18 inserted into this gap during reassembly. Accordingly, a receiving station according to the  
19 invention reduces an amount of time left for receipt of the missing data fragment and reassembly  
20 of the datagram.

1 Reducing the remaining reassembly time to eight seconds in such a situation has  
2 been found to produce good results. Eight seconds has been found generally to allow enough  
3 time for receipt of a data fragment that has been merely delayed, while generally not allowing  
4 enough time for transmission of another datagram with the same IP identification number as the  
5 datagram with the gap.  
6

7 Of course, if the remaining time before timeout is less than eight seconds, only the  
8 remaining time is allowed before timeout. In other words, the remaining time is not increased to  
9 eight seconds if it is already less than eight seconds.  
10

11 By virtue of the foregoing operation, fewer opportunities for datagram  
12 misassembly tend to occur.  
13

14 Fig. 7 is a representational view of a receiving station reducing a remaining time  
15 for reassembling a datagram upon detection of a data fragment from another datagram having  
16 that same source address and protocol as the datagram but a different IP identification number.  
17

18 Briefly, a likelihood of misassembly of data fragments from fragmented IP  
19 datagrams is reduced. Data fragments of a first datagram are received, with the datagram having  
20 a protocol identification number, a source address, and a first IP identification number. A  
21 remaining time for reassembling the first datagram is reduced upon detection, before receipt of a

1 last data fragment of the first datagram, of a data fragment of a second datagram having the  
2 protocol identification number and the source address but having a second IP identification  
3 number. Preferably, this technique is performed by an IP layer of a receiving station's  
4 communication system.

5  
6 In more detail, one problem with attempting to detect a gap in data fragments is  
7 that IP does not provide enough information to directly detect loss of a last data fragment or  
8 fragments of a datagram. In particular, IP data fragments indicate if they are or are not a last data  
9 fragment. The only indication of relative positions of intermediate IP data fragments are offsets  
10 from a start of the datagram. These offsets provide no information about how many data  
11 fragments follow a given data fragment. Thus, if a last data fragment of a first datagram is lost, a  
12 receiving station only knows that it has received some data fragments of the first datagram but  
13 has not yet received a last data fragment. In this situation, the receiving station might receive  
14 before timeout a data fragment from another datagram that happens to match the first datagram's  
15 IP identification number. This occurrence can lead to misassembly of the first datagram.

16  
17 Typically, a sending station will send all of a datagram in a particular protocol to a  
18 particular receiving station before sending another datagram in that protocol to that receiving  
19 station. Thus, possible loss of a last data fragment of a first datagram can be indicated by receipt  
20 of a data fragment from a second datagram sent by the same sending station as the first datagram  
21 using the same protocol. A receiving station can tell that the data fragment is from the second

1 datagram by checking for a different IP identification number than that used by the first  
2 datagram. Accordingly, in order to help prevent misassembly, a sending station can reduce a  
3 time remaining for reassembling a datagram upon receipt of a data fragment from another  
4 datagram having that same source address and protocol as the datagram but a different IP  
5 identification number.

6  
7 Accordingly, receiving station 3 in Fig. 7 has received data fragments A1, A2, A3  
8 and A4 of datagram A, but not last data fragment A5. Receiving station 3 has no way of  
9 knowing if data fragment A5 is a last data fragment of datagram A. Receiving station 3 has  
10 subsequently received data fragment C1 of datagram C. Datagrams A and C have different IP  
11 identification numbers. Therefore, no overlapping can occur. Also, data fragments from  
12 datagram C will not be misassembled with data fragments from datagram A (barring other  
13 processing errors).

14  
15 However, because datagram C shares the same source address and protocol as  
16 datagram A, it is likely that the last data fragment or fragments of datagram A have already been  
17 sent to receiving station 3 and may be lost. Therefore, receiving station 3 according to the  
18 invention preferably reduces a remaining time for receipt of the last data fragment or fragments  
19 and reassembly of the datagram. Reducing the remaining time to eight seconds has been found  
20 to produce good results, allowing for receipt of merely delayed data fragments while still tending  
21 to prevent misassembly.



1           The foregoing methods are designed to decrease a likelihood of datagram  
2 misassembly. Even if misassembly occurs, UDP and other checksums probably will catch most  
3 of the misassembled packets. However, a misassembled packet eventually might slip past the  
4 checksums, especially if only UDP checksums are used, with possibly dire consequences for data  
5 integrity. Thus, the invention also provides techniques for detecting when a likelihood of  
6 datagram misassembly is high so that appropriate corrective action can be taken.

7  
8           Fig. 8 is a flowchart for explaining determination that a likelihood of misassembly  
9 of datagrams is high upon detection of a high rate of communication errors for a period of time.

10  
11           Briefly, a likelihood of misassembly of data fragments from fragmented IP  
12 datagrams is detected. In order to detect this likelihood, communication errors between a  
13 sending station and a receiving station are detected. The likelihood of misassembly is  
14 determined to be high upon detection that the communication errors occur at a high rate for a  
15 predefined period of time.

16  
17           In more detail, step S801 in Fig. 8 detects if communication errors are occurring  
18 at a high rate for a period of time. This error detection can be performed at a sending station or at  
19 a receiving station, both with respect to the station itself and with respect to other stations.

1           The types of errors indicative of datagram misassembly include IP layer  
2 overlapping errors, IP layer timeout, UDP length errors, UDP checksum errors, and NFS errors.  
3 Other errors also might be indicative of datagram misassembly.  
4

5           IP layer overlapping errors can be flagged by a station's IP layer when overlapping  
6 occurs as discussed above with respect to Figs. 3 and 4. Likewise, IP timeout errors can be the  
7 result of reduced timeout and reassembly times as discussed above with respect to Figs. 5, 6 and  
8 7. UDP length and checksum errors can be the direct result of datagram misassembly that is  
9 properly caught by UDP error checking mechanisms. NFS errors can be the result of  
10 misassembly errors that slipped through the IP and UDP error checking mechanisms. NFS errors  
11 can be implied from an increased rate of NFS re-transmissions.  
12

13           Datagram misassembly and situations that create an opportunity for datagram  
14 misassembly have been found to create sustained higher rates of one or more of these types of  
15 errors. Thus, if such errors are detected, flow proceeds from step S801 to S802, and it is  
16 determined that a likelihood of datagram misassembly at the associated receiving stations is high.  
17

18           With a high likelihood of datagram misassembly comes an increased chance that a  
19 misassembled datagram will pass UDP's weak checksum. Accordingly, flow preferably  
20 proceeds from step S802 to step S803, where policies are implemented to decrease the likelihood  
21 of datagram misassembly and to increase a likelihood of catching misassembled datagrams.

1           Examples of the policies implemented in step S803 include using TCP instead of  
2   UDP, if possible. TCP avoids reliance upon IP fragmentation. In addition, additional checksums  
3   can be used. These additional checksums can include application checksums that are much  
4   stronger than those typically used by communication protocols, possibly incorporating extremely  
5   strong hashing functions such as MD5 and SH1. If UDP checksumming is turned off, it can be  
6   turned on. NFS, application, and/or other checksums can be utilized, if possible. Furthermore, a  
7   warning can be sent to the system administrators of both the sending and the receiving stations so  
that the source of the errors can be tracked down and corrected.

8  
9  
10           Some of these policies may not be possible to implement in every situation. For  
11   example, if a server is implementing the policies, the server may not be able to dictate use of  
12   additional checksums to a client. Likewise, TCP may not be available between a particular  
13   sending station and a particular receiving station. In these situations, policies preferably are not  
14   implemented that prevent communications. Of course, if data integrity is essential,  
15   communications with a station that is experiencing high error rates can be discontinued.

16  
17           Fig. 9 is a flowchart for explaining determination that a likelihood of misassembly  
18   of datagrams is high upon determination that an IP identification number generator wraps around  
19   at faster than a predetermined rate.  
20

1 Briefly, a sending station detects a likelihood of misassembly of data fragments  
2 from fragmented IP datagrams sent to a receiving station. In order to detect this likelihood, the  
3 sending station determines a rate at which an IP identification number generator associated with  
4 the receiving station wraps around. The likelihood of misassembly at the receiving station is  
5 determined to be high upon determination that the IP identification number generator wraps  
6 around at faster than a predetermined rate.

7  
8 In more detail, in step S901 of Fig. 9, a sending station determines a rate at which  
9 its IP identification number generator(s) wrap around. In step S902, it is determined if this rate  
10 exceeds a predetermined threshold for any particular IP identification number generator. A  
11 predetermined threshold of 90 seconds has been found to work well.

12  
13 If an IP identification number generator wraps around at faster than the  
14 predetermined rate, a possibility exists that two datagrams having the same IP identification  
15 number will be sent to a receiving station before the first of the two datagrams times out, thereby  
16 creating an opportunity for datagram misassembly. Accordingly, if the threshold is exceeded,  
17 flow proceeds to step S903. In step S903, it is determined that a likelihood of datagram  
18 misassembly at the associated receiving stations is high.

19  
20 Flow then proceeds to step S904, where policies are implemented to address the  
21 high likelihood of datagram misassembly. For example, if the sending station is utilizing plural

1 IP identification number generators, the association between the number generators and the  
2 receiving stations can be changed so that fewer receiving stations are associated with the number  
3 generator that is wrapping around too quickly. With reference to Fig. 2 above, one technique of  
4 changing this association is to change hash 21.

5  
6 In addition, policies along the lines of those discussed above with respect to step  
7 S803 in Fig. 8 also can be implemented. Again, unless data integrity is essential, only those  
8 policies that can be implemented without discontinuing communications preferably are  
9 implemented.  
10  
11

#### 12 *Alternative Embodiments*

13 Each of the techniques discussed above can be used in conjunction with the  
14 others. For example, a sending station can check for communication errors in conjunction with a  
15 high rate of IP identification number generator wraparound. Other combinations of the foregoing  
16 techniques are possible. Thus, while preferred embodiments of the invention are disclosed  
17 herein, many variations are possible which remain within the content, scope and spirit of the  
18 invention, and these variations would become clear to those skilled in the art after perusal of this  
19 application.

20 //

21 //